**Amendments to the Claims:**

This listing of claims will replace all prior versions, and listings, of claims in the application:

**Listing of Claims:**

Claim 1 (currently amended): A method for supporting virtual machines in a data processing system, the method comprising:

automatically determining whether a guest instruction of a guest virtual machine (VM) of a processing system is to be patched for emulation, based at least in part on a list of instructions to be patched, the list of instructions stored in a template map that further stores code templates;

retrieving a code template from the template map that corresponds to the guest instruction to be patched, the code template including an instruction template;

generating an emulation routine for emulating the guest instruction, including modifying the code template based on characteristics of the guest instruction, storing the emulation routine in an emulation database, assigning a value to the emulation routine, and storing an association of the emulation routine with the value in an emulation map;

executing an emulation patch for the guest instruction ~~a guest virtual machine (VM) of a processing system~~, the emulation patch including data to facilitate identification of [[a]] the emulation routine for emulating [[a]] the guest instruction;

in response to execution of the emulation patch, transferring control from the guest VM to a virtual machine monitor (VMM) without saving contextual data that defines a system state for the guest VM; and

using the data from the emulation patch to access the emulation map to find [[an]] the emulation routine for the guest instruction.

Claim 2 (currently amended): A method according to claim 1, wherein the operation of executing an emulation patch comprises executing an instruction that includes an immediate value to be used for finding the emulation routine, wherein instruction replaces a second instruction including a register and not including the immediate value.

Claim 3 (original): A method according to claim 1, wherein the operation of executing an emulation patch comprises executing a flow control instruction, wherein the flow control

instruction includes an address to be used for finding the emulation routine, the flow control instruction selected from a group consisting of:

a call instruction;

a jump instruction; and

a branch instruction.

Claim 4 (original): A method according to claim 1, wherein the operation of executing an emulation patch comprises executing an instruction selected from the group consisting of:

a break instruction;

a branch instruction;

a call instruction; and

a jump instruction.

Claim 5 (currently amended): A method according to claim 1, further comprising:

determining an index, based at least in part on data produced by the emulation patch; and

using the index into the emulation map to find the emulation routine to be executed.

Claims 6-8 (canceled):

Claim 9 (original): A method according to claim 1, further comprising:

automatically determining whether the guest instruction is to be patched for emulation, based at least in part on a list of instructions to be patched, wherein the guest instruction resides in a slot of an instruction bundle;

retrieving a code template that corresponds to the guest instruction to be patched; and

generating the emulation routine for emulating the guest instruction, based at least in part on the code template and on the slot containing the guest instruction.

Claim 10 (previously presented): A method according to claim 1, further comprising:

in response to execution of the emulation patch, finding and executing the emulation routine for the guest instruction without decoding the guest instruction.

Claim 11 (previously presented): A processing system to support virtual machines, the processing system comprising:

a processor;

a machine-accessible medium responsive to the processor; and

instructions in the machine accessible medium, wherein the instructions, when executed by the processing system, cause the processing system to perform operations comprising:

automatically determining whether a guest instruction of a guest virtual machine (VM) of the processing system is to be patched for emulation, based at least in part on a list of instructions to be patched, the list of instructions stored in a template map that further stores code templates;

retrieving a code template from the template map that corresponds to the guest instruction to be patched, the code template including an instruction template;

generating an emulation routine for emulating the guest instruction, including modifying the code template based on characteristics of the guest instruction, storing the emulation routine in an emulation database, assigning a value to the emulation routine, and storing an association of the emulation routine with the value in an emulation map;

executing an emulation patch for the guest instruction ~~a guest virtual machine (VM) of the processing system~~, the emulation patch including data to facilitate identification of [a] the emulation routine for emulating [[a]] the guest instruction;

in response to execution of the emulation patch, transferring control from the guest VM to a virtual machine monitor (VMM) without saving contextual data that defines a system state for the guest VM; and

using the data from the emulation patch to access the emulation map to find [[an]] the emulation routine for the guest instruction.

Claim 12 (original): A processing system according to claim 11, wherein the emulation patch comprises an instruction with an immediate value, the immediate value to be used for finding the emulation routine.

Claim 13 (original): A processing system according to claim 11, wherein the emulation patch comprises a flow control instruction with an address to be used for finding the emulation routine, the flow control instruction selected from a group consisting of:

a call instruction;

a jump instruction; and

a branch instruction.

Claim 14 (original): A processing system according to claim 11, wherein the emulation patch comprises an instruction selected from the group consisting of:

a break instruction;

a branch instruction;

a call instruction; and

a jump instruction.

Claim 15 (currently amended): A processing system according to claim 11, wherein the instructions perform operations comprising:

determining an index, based at least in part on data produced by the emulation patch; and

using the index into the emulation map to find the emulation routine to be executed.

Claims 16-18 (canceled):

Claim 19 (original): A processing system according to claim 11, wherein the instructions cause the processing system to perform operations comprising:

in response to execution of the emulation patch, finding and executing the emulation routine for the guest instruction without decoding the guest instruction.

Claim 20 (currently amended): An apparatus to support virtual machines, the apparatus comprising:

a tangible machine accessible medium; and

instructions in the tangible machine accessible medium, wherein the instructions, when executed by a processing system, cause the processing system to perform operations comprising:

automatically determining whether a guest instruction of a guest virtual machine (VM) of the processing system is to be patched for emulation, based at least in part on a list of instructions to be patched, the list of instructions stored in a template map that further stores code templates;

retrieving a code template from the template map that corresponds to the guest instruction to be patched, the code template including an instruction template;

generating an emulation routine for emulating the guest instruction, including modifying the code template based on characteristics of the guest instruction, storing the emulation routine in an emulation database, assigning a value to the emulation routine, and storing an association of the emulation routine with the value in an emulation map;

executing an emulation patch for the guest instruction ~~a guest virtual machine (VM) of the processing system~~, the emulation patch including data to facilitate identification of [[a]] the emulation routine for emulating [[a]] the guest instruction;

in response to execution of the emulation patch, transferring control from the guest VM to a virtual machine monitor (VMM) without saving contextual data that defines a system state for the guest VM; and

using the data to find an emulation routine for the guest instruction.

Claim 21 (original): An apparatus according to claim 20, wherein the emulation patch comprises an instruction with an immediate value, the immediate value to be used for finding the emulation routine.

Claim 22 (original): An apparatus according to claim 20, wherein the emulation patch comprises a flow control instruction with an address to be used for finding the emulation routine, the flow control instruction selected from a group consisting of:

a call instruction;

a jump instruction; and

a branch instruction.

Claim 23 (original): An apparatus according to claim 20, wherein the emulation patch comprises an instruction selected from the group consisting of:

a break instruction;

a branch instruction;

a call instruction;

a jump instruction.

Claim 24 (currently amended): An apparatus according to claim 20, wherein the instructions perform operations comprising:

determining an index, based at least in part on data produced by the emulation patch; and

using the index into the emulation map to find the emulation routine to be executed.

Claim 25 (currently amended): An apparatus according to claim 20, wherein the instructions perform operations comprising:

~~automatically determining whether the guest instruction is to be patched, based at least in part on a list of instructions to be patched; and~~

inserting the emulation patch in response to a determination that the guest instruction is to be patched.

Claim 26 (canceled):

Claim 27 (original): An apparatus according to claim 20, wherein the instructions, when executed, cause the processing system to perform operations comprising:

in response to execution of the emulation patch, finding and executing the emulation routine for the guest instruction without decoding the guest instruction.